Objective:

The goal of this project is to develop innovative network control software that will significantly enhance the Internet's capability to provide guaranteed and differentiated quality of services (QoS).

Approach:

To understand the SAAM approach, consider road traffic monitoring and control during commute hours in a large city such as San Francisco. In this case, radio stations are the main management entities. The stations send out helicopters to monitor traffic on roads in their respective coverage regions. The information from the helicopters is aggregated at the stations and advice is then broadcast, in real-time, to commuters. The advantages of using these helicopters, which are analogous to the advantages of using SAAM, include: the monitoring of traffic over long routes ("paths") and the early detection of congestion. This monitoring permits advice to be given such as, "it will take about 20 minutes to go from the Bay Bridge to the Civic Center following I-80." In contrast, if the radio stations and helicopters did not exist, each individual motorist could only monitor traffic within a short radius and could not foresee congestion.

Current network management systems are essentially like road traffic monitoring that depends mostly on reports from individual motorists. The SAAM architecture, instead, follows the helicopter model. SAAM achieves timely network management by employing a set of management servers ("helicopters") that we will refer as SAAM servers. These SAAM servers, along with their associated mobile agents, periodically collect state information about the network and maintain "ready to use" path performance data in a Path Information Base (PIB). For scalability, the SAAM servers will be organized in a hierarchy. At the lowest level, each server will maintain a PIB for paths within a small network region. Upper level servers will manage traffic between regions. Additionally, SAAM is designed to facilitate resource reservations for only real-time, long-duration flows such as those carrying video, audio, and large data sets. SAAM will not change how current routers handle best-effort traffic. Thus, the number of flow requests that a SAAM server needs to process will be easily manageable. The SAAM servers form a logical overlay network between the management station and the physical network. As a result, the routers are relieved of routing and network management tasks that they were never designed to perform.

The information stored in the PIB built by SAAM is much more comprehensive than solutions arrived at using shortest path algorithms. In particular, a SAAM server will maintain in its PIB information about all valid paths in the server's region. Such an approach has a couple of advantages. First, it provides efficient support for adaptive routing strategies in which a different routing algorithm (criterion) can be chosen under different network conditions. Second, it significantly increases the probability of the seamless re-routing of real-time flows.

The SAAM servers and the routers, which are designed to enhance both Differentiated and Integrated Services, will use a real-time transport protocol for signaling messages

and agent deployments. Therefore, network management and control tasks will be performed in a timely fashion. Only a small number of planning tasks will require human interactions, and the interactions will be limited to the top-level server in a local administrative domain. SAAM will also have built-in mechanisms to interact with a reservation protocol (such as RSVP) and provide it useful path information when requested.

Unlike a telephone network, the Internet consists of many independently operated ISPs. SAAM can only be progressively deployed. However, this should not be a problem. The SAAM system will be operated by an independent entity that provides value-added services to those ISPs that have joined SAAM by installing a local SAAM server and subscribing to an upper level SAAM server. The ISPs that use SAAM services will provide more predictable performance to their flows thus attracting more customers. Moreover, by using SAAM's efficient resource allocation methods, an ISP will be able to accommodate more traffic and thus collect more revenues. SAAM will support existing inter-domain routing protocols. Therefore, whether or not an ISP is using SAAM is transparent to other ISPs. A non-SAAM ISP can still send traffic through a SAAM ISP. Each ISP has total control over the operation of its internal SAAM server. The upper level SAAM server acts like a consultation center providing only performance enhancing advice to each ISP's SAAM server. Only the internal SAAM servers carry out the actual updates to all routers. Therefore, ISPs should have many incentives to use SAAM.

Recent FY-00 Achievements:

I. Basic research on key SAAM components

(1) Self-repairing Signaling Channels: A two-way signaling channel must be established between a SAAM server and each router under its control. These channels must guarantee reliable and timely delivery of signaling messages. Such performance guarantees must be fault tolerant, i.e., unaffected by changes in network topology caused by device or link failures. This fault-tolerance requirement could be trivially met by flooding each signaling message. However, the overhead incurred by flooding would be too great. Hence, a key requirement for SAAM is the ability to reconfigure the signaling channels automatically and in near real time to accommodate changes in network topology. Legacy routing protocols, which try to detect topological changes after they occur and disseminate the knowledge of these changes hop-by-hop, are too slow for SAAM signaling channel configuration. We have developed a pro-active approach that refreshes SAAM signaling channels over short time intervals in anticipation of topological changes. The overhead of the resulting protocol is very manageable. On average, each router needs to process two control messages in each refresh cycle. The protocol also provides a means for each router to periodically report its link state information to the server without imposing additional processing overhead on the intermediate routers.

(2) Intelligent Resource Manager: The effectiveness of a network resource management scheme depends predominantly on two factors: (1) how complete and

accurate a picture it can obtain about the current state of the network, and (2) how much processing capacity it can use. SAAM offloads resource management processing from routers to a small number of servers. It is economically feasible to design large processing capacity into these servers. Therefore, SAAM has opened a venue for more sophisticated network resource schemes than are currently available. Based on this observation, we have focused on optimality rather than complexity in designing the resource manager to run on a SAAM server. The resulting system supports all service classes defined by major Internet service models (Integrated Services, Differentiated Services, and Multi-Protocol Label Switching) in a cohesive manner. It maintains a comprehensive path information base to aid QoS routing and rerouting and optimizes the utilization of network resources via adaptive routing and dynamic link provisioning between service classes.

(3) Server Fault Tolerance: We have investigated how to make SAAM services tolerant of server failures. There are two types of server failures. Most are transient and recoverable like component failures. The others are catastrophic failures, not recoverable in a short time. For the first type, we examined current commercial offerings and concluded that several of these might be suitable for SAAM. The second type of failures can best be dealt with by using a backup server. No commercial product meets the stringent requirement of SAAM service availability. We have developed a protocol that can detect server failure and resume full service within fractions of a second. The primary and backup servers maintain separate signaling channels with the routers. Both servers collect link state information concurrently. Only the primary server will process and respond to requests for resources. The backup server uses adaptive polling, with the cycle time becoming persistently smaller with each unanswered probe, to detect and verify primary server failure in a timely and reliable manner.

(4) System Security: Security is particularly important for SAAM because SAAM uses mobile code, called resident agents, to extend router services. The server loads these resident agents onto routers dynamically, and the agents then execute on the destination routers. A scheme to authenticate mobile code is required to prevent an outsider from installing a malicious resident agent. Also, all signaling messages in SAAM are authenticated to counter spoofing attacks. The idea of Time-driven Key Sequencing (TKS) has been explored for speeding up the authentication process. TKS is a scheme to implement low-overhead key changes in support of the use of efficient cryptographic algorithms. The general notion is that with frequent key changes, more efficient, but less time-durable, cryptographic algorithms may be utilized to provide an equivalent level of protection compared to the use of more time-durable algorithms with long-term keys. A Kerberos based method is also designed to authenticate new nodes that join a SAAM network.

(5) Server Originated Probing: The objective of this work is to add server-based, router performance sampling capabilities to SAAM. As a router may be misconfigured, or worse, actively attacked, a server should not rely entirely on link performance data reported by routers to maintain the network status. The server must have an independent means to validate link performance reports from a router so that erroneous performance

data can be filtered out before it causes severe service degradation. Thus, the server launches a probe session by first identifying a data path that goes through the target router on a designated port. The server then injects a special agent into the two routers that are upstream and downstream of the target with respect to that path. The agent installed on the upstream router will create measurement packets and forward them via the selected data path. These measurement packets are given a header used by an actual application flow. Therefore, the router under probe will not be able to detect them. The agent installed on the downstream router will extract these measurement packets based on a special payload value and collect their performance data (average delay, loss rate, and throughput). Finally the performance data will be delivered to the server by the downstream router.

(6) Path-based network Policy Language (PPL): Existing network policy languages define policy rules on a per node basis. PPL's path-based approach for representing network policies is advantageous in that QoS and security policies can be associated with an explicit path through the network. This assignment of policies to network flows aids in new initiatives such as Integrated Services. The more stringent requirement of supporting path-based policies can be easily relaxed with the use of wild card characters to also support Differentiated Services and best-effort service. Path-based policies have a complexity advantage over node-based ones as well. When a policy server associates policies with nodes rather than paths, a valid path must be constructed for each new request. This construction not only uses node connectivity information to build the possible paths, but applies the policy information from each node as well. If the path generation request involves a combination of service constraints, such as minimum-delay and least-cost, it becomes an NP-complete problem. Path-based policy is analogous to the use of static routes. Rather than calculating a route through the network, a valid route is specified ahead of time. This pre-specified route accelerates the routing process. When a path is specified ahead of time with the proper policy constraints, this too will accelerate the response to a path request.

II. Improvements on SAAM Testing Environment

(1) Configuration Management: We have simplified SAAM testbed configuration management. A formal test configuration language is defined using XML. A GUI based application is provided to help users create test configurations in the defined language. The demo-station has been modified to set up a SAAM testbed based on a test configuration file.

(2) Integration: The signaling channel configuration protocol, the backup server functionality, the server probing capability, and an initial version of the resource manager have been integrated into the current SAAM prototype.

Results from FY00 are documented on the SAAM public web page, as well as in the proceedings of peer-reviewed conferences and workshops. More than 10 MS theses and one PhD dissertation have been or are being produced based on the results.

FY-01 Plans:

The short-term objective for FY-01 is to make the first public release of SAAM. The code will be released via the Web. Most of the development work planned for this release has been completed. The major tasks remaining are integration, testing and documentation.

More research is also planned to enhance various SAAM components. The work is composed of the following tasks:

(1) A formal performance analysis will be conducted for the server-based resource manager. The embedded resource management algorithm will be fine-tuned based on the performance analysis as well as test results. The advantages of adaptive routing and dynamic link provisioning will be quantified.

(2) More analysis will be conducted to validate the security system that has been designed for SAAM. The complete system will be implemented and tested.

(3) A compiler will be developed for PPL. It will be able to detect and resolve conflicts between polices. PPL will be enhanced to allow a policy maker to specify elaborate procedures for conflict resolution and heuristics to reach a compromise.

(4) The server originated probing functionality will be integrated with the resource manager. The open questions that need to be answered for the integration include when to probe and how to use the probing results.

(5) Most work done so far is for a single SAAM region. The results will be extended to multiple regions.

[Geoff: If time and resource permit, a kernel-level router that replaces the current user-level router emulation will be designed and built to facilitate realistic performance tradeoff studies.]

Technology Transition:

All SAAM system code, including the SAAM testing environment, will be released to the public. A Web site will be developed to disseminate the code and encourage feedback from users. (Several development and research groups have already expressed interest in experimenting with components of the SAAM system.) We expect that the typical users will develop new applications (or fine-tune existing ones) that will take advantage of the differentiated and integrated services capabilities of future networks. Other users may wish to use the SAAM testing environment to rapidly prototype and test their own server-based network management algorithms.

Additionally, the NASA Ames Research Center has a need for a centralized network policy management and enforcement system. To this end, they have expressed an interest in participating in an alpha test of some of the SAAM components.